Name: Meklit Melkamu

ID: ETS1029/14

SECTION:C

1. Identify the types of program flow control and provide:

A) List all flow controls

Sequential statements: follows a serial or sequential flow in which the flow depends on the series of instructions given to the computer. In sequential flow of control, the statements of a program are executed from top to bottom in the order in which they are written.

Selection (conditional)Statements: Selection or conditional flow control: Selection Logic simply involves a number of conditions or parameters which decides one out of several written modules. They perform different computations or actions depending on whether a programmer-specified Boolean condition evaluates to true or false

They are subdivided into: - One-way selection (if statement)

Two-way selection (if...else statement)

Multiple selection (nested if)

Switch statement

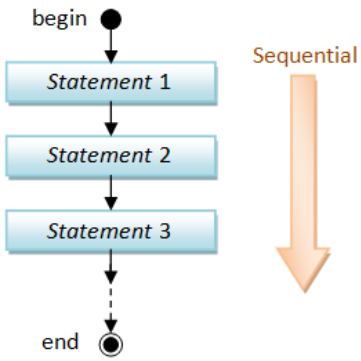
Loop(iteration) Statements: The Iteration logic employs a loop which involves a repeat statement followed by a module known as the body of a loop. A loop is a sequence of statements which is specified once but which may be carried out several times in succession.

They are subdivided into: - For loop

Do while loop

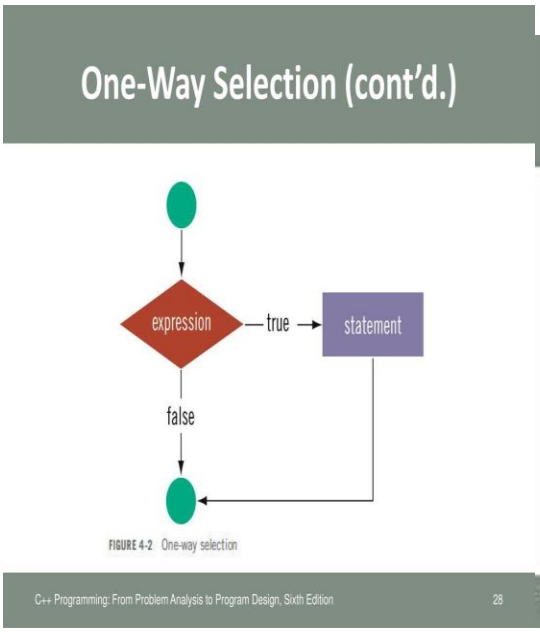
While loop

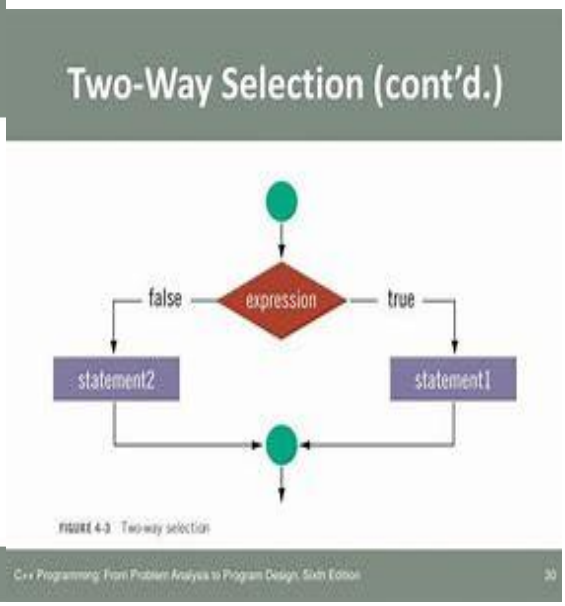
B) Flow chart for each flow controls:



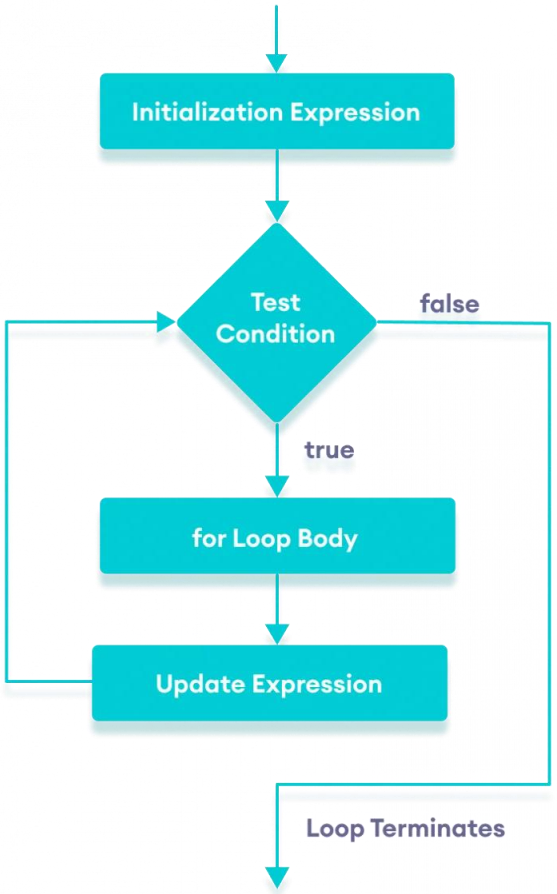
Sequential flow control

Selection flow control

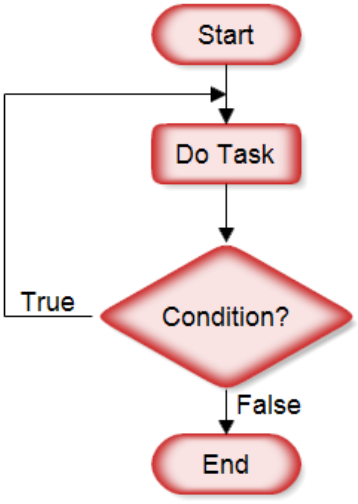




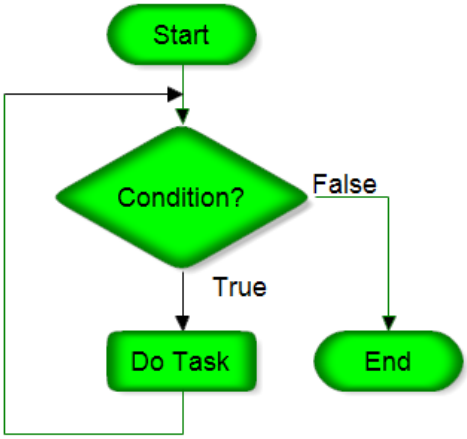
Loop flowcontrol



Do While Loop



While Loop



C) Syntax along with description of the parts,

- one-way selection is: **if (expression)**

  {statement;}

Note that the parentheses around the expression are part of the syntax. The expression is sometimes called a decision maker because it decides whether to execute the statement that follows it. The expression is usually a logical expression.

- Two-way selection: **if (expression) {statement1;}**

  else {statement2;}

Note that Statements 1 and 2 are any valid C++ statements. In a two-way selection, if the value of the expression is true, statement1 executes. If the value of the expression is false, statement2 executes. Figure 4-3 shows the flow of execution of the if...else statement (two-way selection).

- Multiple- way selection: **if (condition1) {statement1;}**

  else if (condition2) {statement 2;}

  else {statement3;}

- Switch statements: **switch (expression) {**

  case value1: statements1

  break;

  case value2: statements2

  break;

  .

  .

  case valuen: statementsn

  break;

  default: statements}

Note In a switch structure, first the expression is evaluated. The value of the expression is then used to perform the actions specified in the statements that follow the reserved word case.

- For loop: **for (initial statement; loop condition; update statement) {**

  Statement;}

Note The initial statement initializes our variable which is a process of giving our variable a starting value. The update statement is a way of increasing and decrementing as it repeats our statement again and again.

- While loop: **while (expression)**

  {statement;}

Note while loops can be counter-controlled, sentinel-controlled, flag-controlled and EOF-controlled.

- Do-While loop: **do {**

  statement;}

  while (expression);

Note The statement executes first, and then the expression is evaluated. If the expression evaluates to true, the statement executes again. As long as the expression in a do...while statement is true, the statement executes.

D) Usage: Here's an example problem where flow control can be used:

Problem: Write a program that asks the user to input a number, and then prints all the even numbers from 2 to that number.

Solution: We can use a while loop and an if statement to check if each number is even. **#include<iostream> using namespace std; int main() { int n; //for user input cout << "Enter a number: "; cin >> n; int i = 2; //start with the first even number while(i <= n) { if(i % 2 == 0) { //check if i is even cout << i << " "; } i++; //increment i to check the next number } return 0; }** In this code, the while loop runs from 2 to the user-inputted number. The if statement inside the loop checks if a number is even by using the modulo operator `%` to check if the remainder of dividing by 2 is 0. If the number is even, it is printed to the console. The code then increments `i` to check the next number. This process continues until all even numbers from 2 to `n` have been printed.

2. What are the operators used to form expressions for flow controls?

Comparison operators: == (equals) , != (not equals) ,< (less than), > (greater than), <= (less than or equal to), >= (greater than or equal to)

Logical operators: && (logical AND), || (logical OR),! (Logical NOT)

Arithmetic operators: + (addition), - (subtraction), * (multiplication), / (division), %(modulus)

Bitwise operators: & (bitwise AND), | (bitwise OR), ^ (bitwise XOR), ~ (bitwise NOT), << (left shift), >> (right shift)

3. Give an example of expression used with flow controls.

```
#include<iostream>
Using namespace std;
 int main() {
 int num1 = 10, num2 = 5;
if (num1 > num2) {
cout << "Num1 is greater than Num2.";}
else {cout << "Num2 is greater than or equal to Num1.";}
return 0;
}
```

Note: If the condition is true, we print the message "Num1 is greater than Num2.", otherwise we print the message "Num2 is greater than or equal to Num1." This is an example of using an expression in a flow control statement to control the flow of a program based on a certain condition**.**